UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 10/600,995 | 06/20/2003 | Ben Jai | 5-4-52 | 5758 |

7590 10/30/2009

Ryan, Mason & Lewis, LLP
Suite 205
1300 Post Road
Fairfield, CT 06824

| EXAMINER | |
|---|---|
| BRUCKART, BENJAMIN R | |
| ART UNIT | PAPER NUMBER |
| 2446 | |

| MAIL DATE | DELIVERY MODE |
|---|---|
| 10/30/2009 | PAPER |

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

**BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES**

Application Number: 10/600,995
Filing Date: June 20, 2003
Appellant(s): JAI ET AL.

Kevin M. Mason, Reg. No. 36,597
For Appellant

**EXAMINER'S ANSWER**

This is in response to the appeal brief filed 8-19-09 appealing from the Office action mailed 10/28/08.

**(1) Real Party in Interest**

A statement identifying by name the real party in interest is contained in the brief.

**(2) Related Appeals and Interferences**

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

**(3) Status of Claims**

The statement of the status of claims contained in the brief is correct.

**(4) Status of Amendments After Final**

The appellant's statement of the status of amendments after final rejection contained in the brief is incorrect.

The amendment after final rejection filed on 12/29/08 has been entered.

**(5) Summary of Claimed Subject Matter**

The summary of claimed subject matter contained in the brief is correct.

**(6) Grounds of Rejection to be Reviewed on Appeal**

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

**(7) Claims Appendix**

The copy of the appealed claims contained in the Appendix to the brief is correct.

**(8) Evidence Relied Upon**

U.S. Patent Publication 20020120720 by Moir, Published 8-2002

U.S. Patent Publication 20010047385 by Tuatini, Published 11-2001

U.S. Patent Publication 20030105838 by Presley, Published 6-2003.

**(9) Grounds of Rejection**

The following ground(s) of rejection are applicable to the appealed claims:

**Claim Rejections - 35 USC § 102**

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

**Claims 1-9, 14-16, 20 and 21 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent Publication No. 20020120720 by Moir.**

Regarding claim 1, in a system having a plurality of devices (Moir: page 4, para 54), wherein a plurality of configuration elements are associated with the plurality of devices (Moir: page 4, para 54-55; each device has component parameters), a method for automated generation of

executable modules associated with the devices (Moir: page 1, para 7; generating rule programs), the method comprising the steps of:

accessing information about one or more input configuration elements of the plurality of configuration elements (Moir: page 4, para 54-55; subcomponents, dependencies), wherein the one or more input configuration elements are associated with one or more input rules (Moir: page 4, para 56-page 5, para 58; rule files and operations file descript operations of components);

determining which of the plurality of configuration elements could be accessed based on the one or more input rules (Moir: page 5, para 57-58; component behavior and constraint definitions);

generating one or more output rules using at least the accessed information, the accessed configuration elements, and the input rules, wherein an output rule corresponds to one or more input configuration elements and wherein said one or more input rules comprise one or more executable statements (Moir: page 5, para 58; the rule program is derived by compiling the rule file and operations file); and

generating at least one executable module adapted to access at least a given one of the input configuration elements and to trigger one or more of the output rules corresponding to the given input configuration element (Moir: page 5, para 60; binding the rule program to "process behavior definitions, labeled operations" for operation via a rule class).

Regarding claim 2. The method of claim 1, wherein the one or more input configuration elements are described by one or more configuration classes (Moir: page 5, para 69- page 6, para 70) and wherein the one or more input rules are described by one or more rule files (Moir: page 5, para 57, 62).

Regarding claim 3. The method of claim 1, wherein the step of determining which of the plurality of configuration elements could be accessed further comprises the step of determining read and write sets of configuration elements for a given one of the one or more rules (Moir: page 9, para 96-98; with overwriting; page 6, para 70; action for writing and predicate for reads).

Regarding claim 4. The method of claim 3, wherein the step of determining read and write sets of configuration elements further comprises the step of determining for the given rule a call chain emanating from the given rule (Moir: page 8, continued rule file from page 7. The component may have multiple operations with the same identifier but with different argument types; calling different operations).

Regarding claim 5. The method of claim 4, wherein the step of determining for a given rule a call chain emanating from the rule further comprises the steps of determining whether the given rule accesses one or more items and determining whether one or more other configuration elements are accessed by the one or more items (Moir: page 8, continued rule file from page 7. invocation of an operation).

Regarding claim 6. The method of claim 5, wherein the one or more items comprise one or more rules or one or more utility methods (Moir: page 5; para 60; number of rules; page 6, para 72; number of rules defined).

Regarding claim 7. The method of claim 5, wherein the step of determining read and write sets of configuration elements further comprises the steps of determining whether the one or more items accesses one or more additional items and determining whether one or more additional configuration elements are accessed by the one or more additional items (Moir: page 8, continued rule file from page 7. invocation of more than operation with different arguments).

Regarding claim 8. The method of claim 1, wherein the step of determining which of the plurality of configuration elements could be accessed further comprise the step of determining, for a given one of one or more configuration elements able to be accessed by an input rule, a set of instance chain accesses for the given configuration element (Moir: page 8, continued rule file from page 7. The component may have multiple operations with the same identifier but with different argument types; calling different operations).

Regarding claim 9. The method of claim 8, wherein the given configuration element comprises a configuration element of a configuration class, wherein the given configuration element is another configuration class (Moir: page 6, para 72; number of rules defined within a rule file), and wherein the step of determining, for a given one of one or more configuration elements able to be accessed by an input rule, a set of instance chain accesses for the given configuration element further comprises the step of determining every access for the other configuration class to other configuration elements (Moir: page 8, continued rule file from page 7. invocation of more than operation with different arguments).

Regarding claim 14. The method of claim 1, wherein the at least one executable module is adapted to trigger the one or more output rules corresponding to the given input configuration element through deferred triggering of the one or more output rules (Moir: page 8; <operation> is similar to the definition in the spec that "deferred rules are invoked as a sequence, typically at the end of a configuration session").

Regarding claim 15. The method of claim 1, wherein the at least one executable module is adapted to trigger the one or more output rules corresponding to the given input configuration element through direct triggering of the one or more output rules (Moir: page 9, para 105).

Regarding claim 16. The method of claim 1, wherein the at least one executable module is adapted to trigger the one or more output rules corresponding to the given input configuration element through batch triggering of the one or more output rules (Moir: page 8, continued rule file from page 7. batch of operations with the same identifier but with different argument types).

Regarding claim 20. In a system having a plurality of devices (Moir: Fig. 12; page 4, para 54), wherein a plurality of configuration elements are associated with the plurality of devices (Moir: page 4, para 54-55; each device has component parameters), an apparatus for automated generation of executable modules associated with the devices (Moir: page 1, para 7; generating rule programs), the apparatus comprising:

a memory (Moir: page 12, para 131); and

at least one processor, coupled to the memory (Moir: page 12, para 131);

the apparatus being operative:

to access information about one or more input configuration elements of the plurality of configuration elements (Moir: page 4, para 54-55; subcomponents, dependencies), wherein the one or more input configuration elements are associated with one or more input rules (Moir: page 4, para 56-page 5, para 58; rule files and operations file descript operations of components);

to determine which of the plurality of configuration elements could be accessed based on the one or more input rules (Moir: page 5, para 57-58; component behavior and constraint definitions);

to generate one or more output rules using at least the accessed information, the accessed configuration elements, and the input rules, wherein an output rule corresponds to one or more input configuration elements and wherein said one or more input rules comprise one or more executable statements (Moir: page 5, para 58; the rule program is derived by compiling the rule file and operations file); and

to generate at least one executable module adapted to access at least a given one of the input configuration elements and to trigger one or more of the output rules corresponding to the given input configuration element (Moir: page 5, para 60; binding the rule program to "process behavior definitions, labeled operations" for operation via a rule class).

Regarding claim 21. An article of manufacture for use in a system having a plurality of devices (Moir: Fig. 12; page 4, para 54; page 12, para 131), wherein a plurality of configuration elements are associated with the plurality of devices (Moir: page 4, para 54-55; each device has component parameters), and for automated generation of executable modules associated with the device (Moir: page 1, para 7; generating rule programs), the article of manufacture comprising:

a machine readable medium containing one or more programs which when executed implement the steps of (Moir: page 12, para 131):

accessing information about one or more input configuration elements of the plurality of configuration elements (Moir: page 4, para 54-55; subcomponents, dependencies), wherein the

one or more input configuration elements are associated with one or more input rules (Moir: page 4, para 56-page 5, para 58; rule files and operations file descript operations of components);

determining which of the plurality of configuration elements could be accessed based on the one or more input rules (Moir: page 5, para 57-58; component behavior and constraint definitions);

generating one or more output rules using at least the accessed information, the accessed configuration elements, and the input rules, wherein an output rule corresponds to one or more input configuration elements and wherein said one or more input rules comprise one or more executable statements (Moir: page 5, para 58; the rule program is derived by compiling the rule file and operations file); and

generating at least one executable module adapted to access at least a given one of the input configuration elements and to trigger one or more of the output rules corresponding to the given input configuration element (Moir: page 5, para 60; binding the rule program to "process behavior definitions, labeled operations" for operation via a rule class).

**Claim Rejections - 35 USC § 103**

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

**Claims 10-13 are rejected under 35 U.S.C. 103(a) as being unpatentable by U.S. Patent Publication No. 20020120720 by Moir in view of U.S. Patent No. 2001/0047385 by Tuatini.**

Regarding claim 10. The Moir reference teaches the method of claim 1.

The Moir reference fails to explicitly teach creating a class but does teach rule classes and creating an executable module and structured output files (Moir: page 5, para 60).

However, the Tuatini reference teaches generating at least one class for a given one of the one or more output rules, the at least one class defining the at least one executable module

(Tuatini: page 8, para 84; page 11, para 108) in order to use a standardized high level language format allowing an easy and cost effective maintenance (Tuatini: page 1, para 4-5).

It would have been obvious to one of ordinary skill in the art to create the method of generating an executable module as taught by Moir to include class file as taught by Tuatini in order to use a standardized high level language format allowing an easy and cost effective maintenance (Tuatini: page 1, para 4-5).

Regarding claim 11. The method of claim 10, wherein the at least one class comprises one or more statements adapted to access at least one given configuration element that corresponds to the one or more output rules (Moir: page 8, para 75; rule program).

Regarding claim 12. The method of claim 10, wherein each of the at least one classes comprises one or more methods adapted to access the at least one given configuration element (Moir: page 8, para 75; op number).

Regarding claim 13. The method of claim 12, wherein the access comprises reading, writing, or modifying the at least one given configuration element (Moir: page 9, para 96-98; with overwriting; page 6, para 70; action for writing and predicate for reads).

**Claims 17-19 are rejected under 35 U.S.C. 103(a) as being unpatentable by U.S. Patent Publication No. 20020120720 by Moir in view of U.S. Patent No. 2003/0105838 by Presley.**

Regarding claim 17. The Moir reference teaches the method of claim 3, wherein the one or more output rules comprise two or more output rules (Moir: page 5, para 58).

The Moir reference fails to teach performing a circularity check.

However, the Presley reference teaches a method further comprises the step of performing a circularity check by determining dependency relationships between the two or more output rules and by determining whether a given one of the two or more output rules depends upon itself (Presley: page 4, para 47) in order to provide reliable and predictable performance (Presley: page 1, para 5).

It would have been obvious to one of ordinary skill in the art at the time of the invention to create the method of claim 3 as taught by Moir to include determining dependency relationships in order to provide reliable and predictable performance (Presley: page 1, para 5).

Regarding claim 18. The Moir reference teaches the method of claim 1.

The Moir reference fails to teach range restriction.

However, the Presley reference teaches, wherein the information further comprises at least one range restriction corresponding to the given input configuration element and wherein the at least one executable module is adapted to ensure that the at least one range restriction is met when the given configuration element accessed by the one or more triggered output rules is assigned a value (Prseley: page 4, para 52) in order to provide reliable and predictable performance (Presley: page 1, para 5).

It would have been obvious to one of ordinary skill in the art at the time of the invention to create the method of claim 3 as taught by Moir to include determining dependency relationships in order to provide reliable and predictable performance (Presley: page 1, para 5).

Regarding claim 19. The Moir reference teaches the method of claim 1.

The Moir reference fails to teach dependency integers.

However, the Presley reference teaches information further comprises at least one referential integrity restriction corresponding to the given input configuration element and wherein the at least one executable module is further adapted to ensure that the at least one referential integrity restriction is met when the given configuration element is accessed by the one or more triggered output rules (Presley: page 4, para 53; referential integrity restriction specifies that a variable is dependent on the state of another variable) in order to provide reliable and predictable performance (Presley: page 1, para 5).

It would have been obvious to one of ordinary skill in the art at the time of the invention to create the method of claim 3 as taught by Moir to include determining dependency relationships in order to provide reliable and predictable performance (Presley: page 1, para 5).

**(10) Response to Argument**

The main issue regarding the independent claims centers around whether Moir teaches input rules as defined by appellant. Appellant cites the specification to define this feature and in the claims defines the feature as input rules comprising one or more executable statements. The definition of executable statements is the subject of such appeal.

**The Appellant Argues:**

1) Appellant argues the Moir reference does not teach the third and fourth steps found in the body of the independent claims.

**In response**, the examiner respectfully submits:
The cited reference Moir teaches the invention as claimed.

Claims 1, 20 and 21 recite in the preamble "a system having a plurality of devices" shown in Moir page 4, para 54 with network devices, "wherein a plurality of configuration elements are associated with the plurality of devices" Moir: page 4, para 54-55 teaches devices are broken down into software and hardware subcomponents and that those components have parameters that are configured for operation with reference to each other and the network. The system claim implements "a method for automated generation of executable modules associated with the devices" shown in Moir: page 1, para 7; generating rule programs. Moir teaches a rule program is generated utilizing the rule file and operations file.

This method comprising the steps of:

accessing information about one or more input configuration elements of the plurality of configuration elements (Moir: page 4, para 54-55; Moir shows that each device has individual subcomponents and that they many components have dependencies on other components complicating network administration and configuration), wherein the one or more input

configuration elements are associated with one or more input rules (Moir: page 4, para 56-page 5, para 58; Moir teaches there are rule files and operations file (Fig. 8, tags 62 and 64). The operations file descript operations of components. Here Moir links the input configuration elements to the input rules);

determining which of the plurality of configuration elements could be accessed based on the one or more input rules (Moir: page 5, para 57-58; The rule file that specifies behavior requirements of a specific device component behavior and the operations file defines constraint definitions).

The third and fourth limitations mentioned and mapped to the reference here are the subject of appellant's arguments.

generating one or more output rules using at least the accessed information, the accessed configuration elements, and the input rules, wherein an output rule corresponds to one or more input configuration elements and wherein said one or more input rules comprise one or more executable statements (Moir: page 5, para 58; the output rule is the rule program which is derived by compiling the rule file and operations file- see Fig. 9); and

generating at least one executable module adapted to access at least a given one of the input configuration elements and to trigger one or more of the output rules corresponding to the given input configuration element (Moir: page 5, para 60; binding the rule program to "process behavior definitions, labeled operations" for operation via a rule class).

Appellant argues, on the bottom of page 5 of the brief, that the examiner asserts that the configurations are executed to determine and control on how a device is to behave. The examiner did state this. As Moir teaches on page 4, para 54-56, the network devices are broken down into components and that "the correct operation of the network devices depends on the correct configuration of component parameters" of these each components. Para 55 explains that configuring a device is a typical administrative task but it tends to lend itself to complex and expensive network administration of "configuring component parameters." The Moir reference seeks to alleviate that problem. By configuring a device, you are inputting and establishing how that device is to behavior in the network and how it is to interact with other devices, as noted by the dependency between parameters and configurations of Moir para 55.

On page 7 ,of the brief, appellant cites this section as proof that configuration files are not executed. The examiner maintains that the configuration files are indeed executed and utilized by the virtual machine compiler to produce the output rules. These configurations files are executed and used by the compiler to generate to the output rules (the rules program) (see Fig. 9). Otherwise, the configurations would not be referenced as dependent upon each other for each network device "to perform a task" and that the configurations define "behavior constraints" and "behavior requirements" of each device.

Appellant argues on the bottom of page 5, top of page 6, that the examiner asserts that "executable statements is broad and is not limited to code or a certain type of statement." This statement is also true. Appellant is seeking to distinguish the executable statements of the input rules (claim 1, step 3) from the Moir operations file and rule file (Fig. 9 and para 57-58). In previous actions, the examiner had requested applicant to further detail these features in the claims but appellant relies on the specification to give narrow and specific definitions of each element (see brief page 6 and previous responses).

In response to applicant's argument that the references fail to show certain features of applicant's invention, it is noted that the features upon which applicant relies (i.e., definitions of input rules, output rules and executable modules from specification pages 2-3) are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

Applicant only defines the input rules as having "one or more executable statements." This is a broad and sweeping definition. Nonetheless, Moir teaches the operations file and rules file contain statements, constraints, dependencies all of which are taken into consideration and executed by the virtual machine compiler when creating the rule program.

The examiner maintains that because the operations file and the rule file are processed by a virtual machine compiler (a form of execution by being processed), they meet the claim limitations. The Moir reference processes and executes the operations file and rule file in order to achieve the output of the rules program. Because the virtual machine compiler creates the rules

program from the operations file and the rules file, the operations and rule files are processed and executed to create the program. An example of the operations file is provided on page 6 of the publication (top half of the page). Para 72 is the most conclusive evidence that the rule has executable statements. Para 72 states "Turning now to the rule file 65, as stated above the rule file 64 is text that is converted into a binary-form rule program 66. Within the rule file 64, in one exemplary embodiment, a number of rules may be defined, each rule comprising a decision tree having the general form: if <predicate> THEN<action>ELSE<action>. Para 73 shows more examples of such executable statements. This statement is a clear and direct example of an executable statement within the rules file.

This is crucial to the examiner's point because on the bottom of page 7, appellant brings in another definition of executable statement as "performs an action." The example gives examples of if/then statements and evaluation of statements. This is the first time appellant has produced a definition for the term. Such definition of executable statements reads directly on para 72 and 73 of Moir.

Therefore, the examiner's interpretation is confirmed by appellants own evidence and the Moir reference is maintained. The operations file and rule file drive and dictate how the rule program is being made. The statements within each of the files are utilized by the processor to create the rule program. The compiler binds the processes behavior definitions and operations to data through compiling the operations files and rule files. The compiler executes the if/then statements to perform actions and generate the output rules.

**The Appellant Argues, on pages 8-9 of the brief:**

2) Appellant argues with respect to claims 4 and 8, that the Moir reference does not teach the limitations.

**In response**, the examiner respectfully submits:

The cited reference Moir teaches the invention as claimed.

Appellant argues that the present application teaches the "call chain can include calls by the rule to other items such as additional rules and utility methods (Page 3, lines 26-28). Again, in response to applicant's argument that the references fail to show certain features of applicant's

invention, it is noted that the features upon which applicant relies (i.e., the defails of the call chain) are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

Regarding claim 4 (dependent on claim 3), Moir page 8 teaches "determining for the given rule a call chain emanating from the given rule" on page 8, continued rule file from page 7. The component may have multiple operations with the same identifier but with different argument types; calling different operations. Moir illustrates a call chain by showing a rule invoking different ACTIONS and operations of other methods (para 72-73) based on the conditional statement. A chain of calls is referenced by the rule file to perform said operations. Specifically the <mnemonic-ident> identifies other operations to invoke. Here the rule file calls another operation/function. Top of page 9 references the corresponding VOP operations file from the rule file. This constitutes a call chain.

Regarding claim 8 (dependent on claim 1), Moir teaches "determining, for a given one of one or more configuration elements able to be accessed by an input rule, a set of instance chain accesses for the given configuration element" on page 8, where a set of instance chain access for the given configuration are illustration on the rule file from page 7. The call chains are the methods and clauses (bottom of page 7) and operations (top of page 7). Specifically the <mnemonic-ident> identifies other operations to invoke.

**The Appellant Argues, on page 9 of the brief:**

3) Appellant argues with respect to claim 9 (dependent on claim 8), that the Moir reference does not teach the limitations of "wherein the given configuration element comprises a configuration element of the configuration class …".

**In response**, the examiner respectfully submits:

The cited reference Moir teaches the invention as claimed.

This is also a new argument, never presented before appeal. The examiner points to Moir page 6, para 72 to show a "given configuration element comprises a configuration element of a configuration class, wherein the given configuration element is another configuration class." The rule file designates rules within the rule file. A class of an element is not defined in the claim.

The class of element is taken to be the name of the network element which would be the name described on page 7, see <heapObject> and <cstring>. The limitation "wherein the step of determining, for a given one of one or more configuration elements able to be accessed by an input rule, a set of instance chain accesses for the given configuration element further comprises the step of determining every access for the other configuration class to other configuration elements" is similar to the limitation of claim 8. Every acess for the other configuration class to other configuration element will be defined by the conditional statement and operations that are designated by name to the VOP file. The call chains are the methods and clauses (bottom of page 7) and operations (top of page 7). Specifically the <mnemonic-ident> identifies other operations to invoke.

**The Appellant Argues, on page 10 of the brief:**

4) Appellant argues with respect to claim 14 (dependent on claim 1) and 16 (dependent on claim 1), that the Moir reference does not teach the limitations of "wherein the at least one executable module is adapted to trigger the one or more output rules corresponding to the given input configuration element through deferred/batch triggering of the one or more output rules".

**In response**, the examiner respectfully submits:

As admitted by applicant, Moir teaches grammar of the body of a rule. As argued above Moir teaches execution of input rules and creation of an output rule (rules program). The executable module is the virtual machine compiler that triggers (creates) the output rules corresponding to the input elements through deferred and direct triggering of the output rules. The condition of deferred triggering is taught by Moir because the "deferred rules are invoked as a sequence, typically at the end of a configuration session" page 8. Moir also teaches this feature because the operations are not triggered until the rules program is loaded into the network connection device (page 5, para 65).

The condition of batch triggering is taught because the rules in program rules are process specific and the plurality of those rules are generated when the virtual machine compiler utilizes the rules and operation files. The rules and operations files (pages 6-8) detail allow many <processDeclarations> and <operationDeclaractions>. When a plurality of these (a batch) are created in the rules program, a batch of rules are triggered and directly created.

**The Appellant Argues, on page 11 of the brief:**

5) Appellant argues with respect to claim 15 (dependent on claim 1), that the Moir reference does not teach the limitations of "wherein the at least one executable module is adapted to trigger the one or more output rules corresponding to the given input configuration element through direct triggering of the one or more output rules".

**In response**, the examiner respectfully submits:

Moir teaches direct triggering on page 9, para 105 where the rule program may be directly loaded into the network device. The executable module is the virtual machine compiler which creates the output rules (rule program) based on input rules (operations file, rules file). The direct triggering can be any number of things. Appellant has given not details about what direct pertains to. Direct triggering on which device, directly in response to what? The examiner views the direct triggering as the output files being directly send to the network device for which they are created to configure. There is no mention of a proxy or intermediary between the execution module and the network device (Fig. 9). The output rules are triggered when they are implemented because they dictate how the device operates. The output rules are directly triggered by the virtual machine compiler at the time of creation. Each input rule is directly

evaluated and the corresponding operation/action is performed as a direct result of the condition evaluation (if statement).

**The Appellant Argues, on page 12 of the brief:**

6) Appellant argues with respect to claim 17 (dependent on claim 3), that the Moir in view of Presley references do not teach the limitations of "performing a circularity check by determining dependency relationships between two or more output rules and by determining whether a given one of the two or more output rules depends upon itself".

**In response**, the examiner respectfully submits:

Moir teaches generating configuration information for network devices. Moir explicitly mentions the complexity because many devices and configurations depend on each other (para 55). Moir also teaches output rules (as argued above). The feature of dependency between configurations (output rules) and determining whether they depend upon itself is taught by the probing service layers on page 4, para 47, "identify cyclical relationships." If an output rule depends upon itself it is in a cyclical relationship (as defined by applications specification, publication para 94). Presley teaches it looks for these cyclical relationships and other errors that can be construed as out-of-bounds or an invalid configuration. Presley teaches the that by analyzing the configurations you can find and correction configuration errors in order to provide reliable and predictable performance (Presley: page 1, para 5).

The examiner maintains that Moir in view of Presley teach the limitations of performing a circularity check on configurations in order to provide reliable and predictable performance (Presley: page 1, para 5). It would have been obvious to one of ordinary skill in the art to create the invention of Moir to include the circularity check as taught by Presley in order to provide reliable and predictable performance (Presley: page 1, para 5).

Further, KSR supports the rationale of borrowing and substituting a well known idea or feature from one invention into a similar invention. Here the well known idea is proof checking configuration files through a circularity check.

**(11) Related Proceeding(s) Appendix**

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

**Conclusion**

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/Benjamin R Bruckart/
Primary Examiner, Art Unit 2446

/Jeffrey Pwu/
Supervisory Patent Examiner, Art Unit 2446

Conferees:

/Jeffrey Pwu/

Supervisory Patent Examiner, Art Unit 2446

/Joseph E. Avellino/

Supervisory Patent Examiner, Art Unit 2458